Capítulo 3: Modelos para especificação de sistemas de software

Especificação

- Objetivos:
  - mostrar as propriedades (atributos) do sistema.
  - criar uma ponte de comunicação entre os diversos tipos de pessoas envolvidas no desenvolvimento.
- O que muda de uma especificação para outra, nos diversos paradigmas $\rightarrow$ nível de abstração aplicado às propriedades do sistema.
- Duas classes gerais de atores:
  - o produtor de um serviço.
  - o consumidor de um serviço.

- Dois processos:
  1. construção de modelos; e
  2. transmissão de mensagens entre grupos de pessoas.
- Especificação
  - **dos requisitos**: cliente e desenvolvedor.
  - **do projeto geral**: projetista e implementador.
  - **do projeto detalhado**: (projeto de módulos) programadores que utilizam o módulo e os que o implementam.

Princípios da especificação

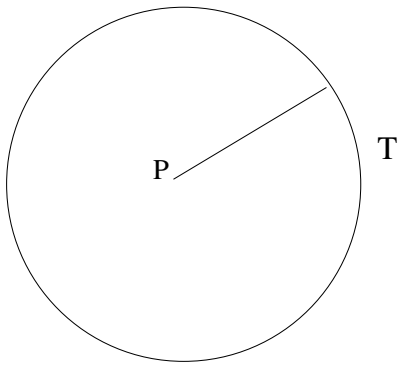
- **Abstração**: filtra apenas aspectos relevantes à fase de desenvolvimento em que se está trabalhando, omitindo informações não pertinentes à fase.
- **Decomposição**: garante que as propriedades do sistema todo sejam refletidas nas propriedades das partes.

Tipos de especificação

- **Operacional**: representa o comportamento desejado do sistema, utilizando modelos abstratos que de alguma forma simulem seu comportamento.
- **Descritiva**: declara as propriedades desejadas do sistema de uma forma puramente descritiva.

A trajetória T de um satélite é um caminho composto de pontos que descrevem seu movimento, de forma que a distância entre cada ponto de T e um ponto fixo P tem um valor constante.

- Operacional:



- Descritiva :

$$x^2 + y^2 + c = 0.$$

- trajetória: conjunto de pontos T;
- coordenadas x e y: satisfazem a equação (ou seja, x e y têm as propriedades desejadas);
- constante c: depende da distância do ponto P aos pontos de T (ou seja, é uma função do raio da circunferência).

Especificação de requisistos

- a partir da declaração de objetivos e restrições do projeto (contrato entre o cliente e o desenvolvedor).
- descrição precisa e não ambígua do comportamento desejado para o sistema, em termos de características funcionais observadas pela interface externa ao sistema (*o que* é esperado do sistema).

- Problemas:
  - O cliente nem sempre tem uma boa visão do domínio da aplicação.
  - As necessidades de clientes ou usuários muitas vezes não são claramente entendidas pelo desenvolvedor.
  - A falta de familiaridade do desenvolvedor com os termos utilizados pelo usuário.

Especificação do projeto

- especificação das características operacionais e da estrutura do sistema (*como* o sistema deve ser implementado para ter o comportamento definido anteriormente).
- deve garantir que o comportamento externo definido no estágio anterior seja preservado (detalhes sobre os dados, ações, controle e execução).

- Três passos:
  1. especificação do projeto geral (decomposição do sistema em subsistemas, definição das relações entre os subsistemas e decomposição dos subsistemas em módulos);
  2. especificação do projeto detalhado (definição da lógica dos módulos); e
  3. especificação das interfaces do sistema.

Especificação de programa

- são escritos programas que devem ser condizentes com o que foi projetado.
- deve-se garantir que as decisões de projeto sejam corretamente traduzidas na especificação dos programas (escolha de melhores algoritmos para a implementação do projeto).

Usos da especificação

- Para auxiliar o usuário a entender suas próprias necessidades e também a validar o produto final.
- Pelo desenvolvedor para tomar decisões durante o projeto e conferir se a implementação está de acordo com a especificação.
- Durante a manutenção para avaliar o impacto das modificações.
- Pelo gerente para controlar o projeto, redirecionando recursos de forma que o projeto tenha custo, tempo e qualidade desejados.

Graus de formalidade

- Usar especificação semiformal durante todo o desenvolvimento.
- Usar especificação formal apenas para representar os requisitos do sistema e então usá-la como guia para escrever o projeto, implementação e testes, utilizando métodos informais ou semiformais.
- Usar especificação formal durante todo o desenvolvimento.

Decisão sobre o grau de formalidade

- propriedades do sistemas (propriedades críticas, como, por exemplo, propriedades de segurança devem ser especificadas de maneira formal).
- componentes do sistema cujo corretismo é crítico ou que foram criados a partir de requisitos vagos deverão ser especificados formalmente.

- formalismos não são aplicados nos ambientes de negócios por consumirem tempo, por serem caros e não serem um bom meio de comunicação entre desenvolvedor e cliente.
- poucos desenvolvedores de software têm experiência suficiente para aplicar métodos formais, sendo necessário utilizar recursos extras em treinamento.

Modelos

- Ferramentas para representar as especificações a ser feitas durante todo o processo de desenvolvimento.
- Representação em miniatura de uma realidade complexa, que reflete certas características específicas do sistema que está sendo representado.
- Úteis se conseguem retratar características relevantes ao sistema.

- Vários modelos do mesmo sistema são usados para examinar diferentes características do sistema em diferentes momentos.
- Podem ser desprezados na construção de sistemas pequenos e/ou pouco complexos.
- Auxiliam na organização de informações e na especificação dos requisitos, mas não na determinação dos requisitos.

Objetivos principais dos modelos

1. representar uma visão do ambiente antes da automação;
2. indicar as diferentes alternativas de solução;
3. apontar as necessidades futuras do sistema, facilitando sua atualização;
4. permitir a avaliação e o refinamento das características do sistema;

5. representar os componentes como partes bem definidas e com dependência mínima entre elas;

6. permitir que se trabalhe gradualmente com a complexidade, partindo-se da visão mais abstrata de um sistema até o seu detalhamento; e

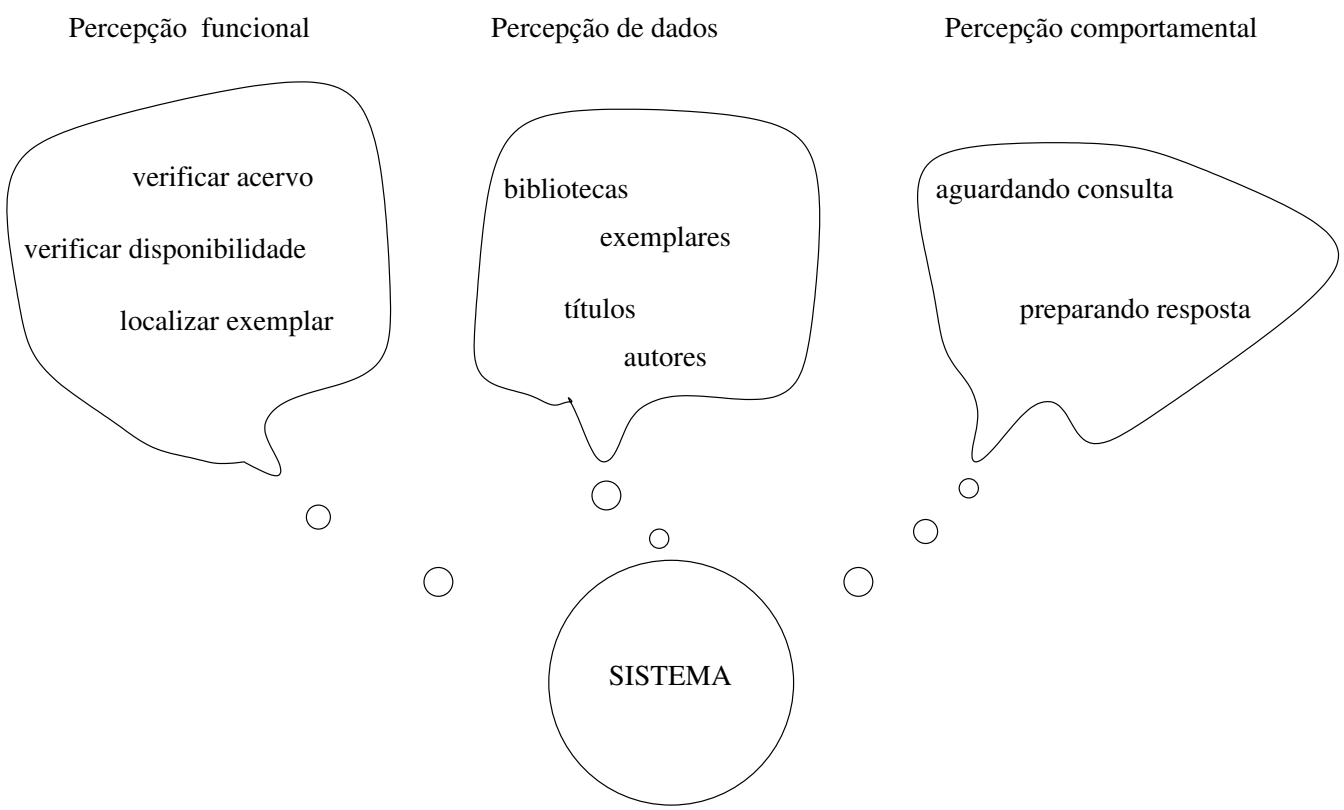
7. fornecer informações quantitativas sobre o escopo e a complexidade do projeto.

Subsistema de consulta a bibliotecas

O subsistema recebe como entrada um título ou um autor. Quando título e autor são fornecidos pelo usuário, o sistema deve consultar o acervo da universidade e verificar se o par título–autor pertence ao acervo. Em caso afirmativo, o sistema deve verificar se há disponibilidade de exemplares correspondentes ao par título–autor e, nesse caso, encontrar a localização desses exemplares. Em caso negativo, o sistema deve informar ao usuário que o par não pertence ao acervo. Quando apenas o título é fornecido, o sistema deve listar todas as ocorrências do título no acervo. Quando apenas o autor é fornecido, o sistema deve informar ao usuário todos os títulos daquele autor pertencentes ao acervo.

Modelo do mundo real

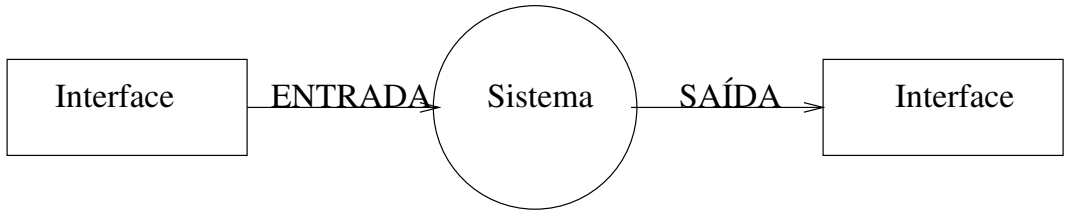
- Descreve a percepção que o desenvolvedor tem do sistema a ser construído
- Focaliza separadamente três características diferentes:
  1. o que o sistema faz;
  2. que dados o sistema mantém; e
  3. como o sistema se comporta.


Percepção funcional
Percepção de dados
Percepção comportamental
verificar acervo
verificar disponibilidade
localizar exemplar
bibliotecas
exemplares
títulos
autores
aguardando consulta
preparando resposta
SISTEMA

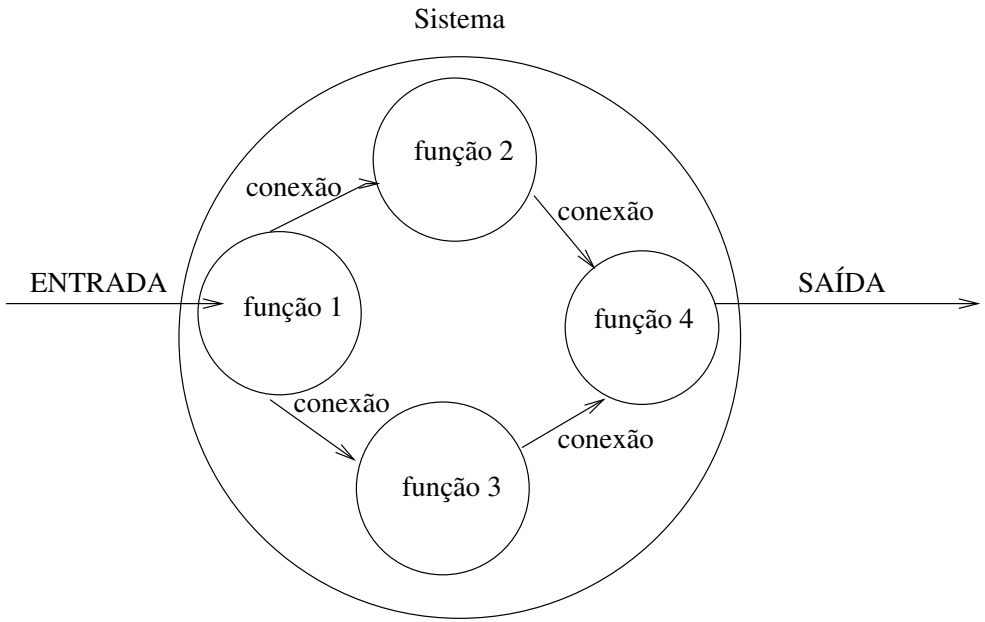
- A percepção do que o sistema *faz* → *modelo funcional do sistema.*
- A percepção dos dados que o sistema *mantém* → *modelo de dados* (ou modelo de informação).
- A percepção de como o sistema se *comporta* → *modelo comportamental*; pode focalizar dados (objetos do sistema) ou funções (importantes principalmente para sistemas de tempo real).

O modelo de função

- **Função**: transformação de dados que entram em dados que saem do sistema.
- O sistema todo é uma função, uma vez que é uma transformação de entradas em saídas.



- O sistema é decomposto identificando-se como componentes suas principais funções.
- A função do sistema todo é constituída por um conjunto de subfunções conectadas.
- Cada conexão representa um duto pelo qual fluem dados.
- Cada uma das subfunções é possivelmente formada por outras subfunções conectadas.
- O modelo funcional do sistema é constituído por uma série de desenhos (rede) representando sucessivas divisões das partes que constituem o sistema.


Sistema
função 2
conexão
conexão
ENTRADA
função 1
função 4
SAÍDA
conexão
conexão
função 3

- Para cada conexão: declaração formal do significado dessa conexão na rede (dados que fluem pelas conexões) colocada em um *dicionário de dados (DD)*.
- O modelo de função está completo quando:
  - descreve todo o sistema, ou seja, mostra as transformações de todas as entradas em saídas;
  - decompõe convenientemente o sistema de modo que todos os componentes não particionados sejam elementares;

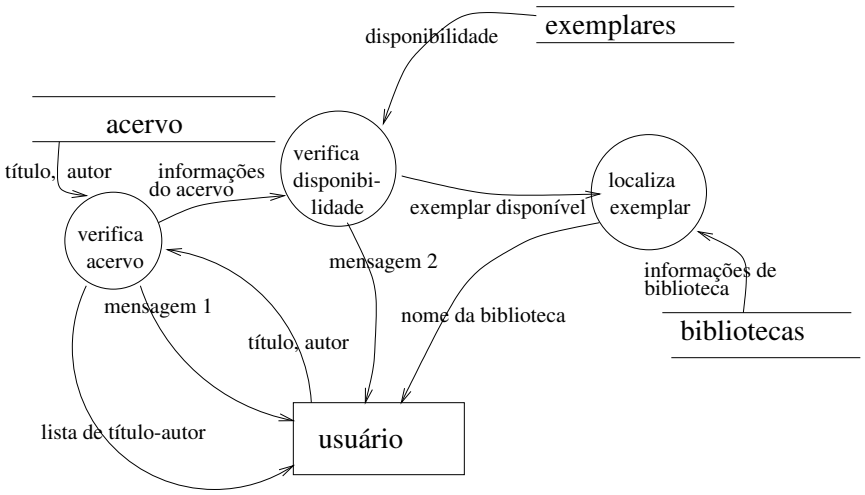
  - cada componente do sistema está ligado corretamente ao resto da rede, e nenhuma conexão necessária foi omitida;
  - as conexões estão minimizadas;
  - cada conexão da rede está bem definida no dicionário de dados, assim como todos os elementos de dados que compõem cada conexão.

- ferramenta mais conhecida para especificação de funções: *diagrama de fluxo de dados* (DFD).

Diagrama de fluxo de dados

- especificação semiformal das funcionalidades descrevendo o sistema como uma coleção de dados que são manipulados por funções (componentes).
- dados podem estar armazenados em *depósitos de dados*, ou contidos em um *fluxo de dados* (conexão) fluindo de uma função para outra, ou sendo transferidos para/do ambiente externo.

- Elementos do DFD:
  - *bolha*: usada para representar função;
  - *seta*: usada para representar conexão, que nesse diagrama é chamada fluxo de dados;
  - *caixa aberta*: usada para representar os depósitos de dados que agrupam os dados mantidos pelo sistema;
  - *caixa retangular*: representa a entidade externa que pode ser origem ou destino dos dados do sistema.


exemplares
disponibilidade
acervo
título, autor
informações
do acervo
verifica
disponibi-
lidade
localiza
exemplar
exemplar disponível
verifica
acervo
mensagem 2
informações de
biblioteca
mensagem 1
nome da biblioteca
título, autor
bibliotecas
lista de título-autor
usuário

- Duas possibilidades de aquisição de dados pela função *verifica acervo*:
  1. informações sobre *título e autor* devem ser fornecidas pelo *usuário* para que a função possa ser executada; ou
  2. a entrada de qualquer uma delas possibilita a execução da função.
- O que ocorre quando uma função recebe várias entradas e produz mais de uma saída?
- Outro aspecto que o DFD deixa de representar: sincronização de componentes de um sistema.

O modelo de dados

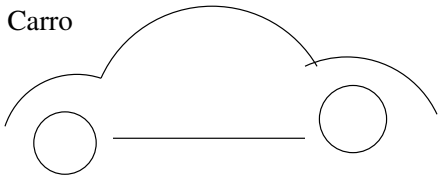
- deve representar:
  - os dados que precisam ser armazenados pelo sistema;
  - a melhor organização desses dados;
  - o relacionamento entre grupos de dados; e
  - como eles serão utilizados.
- representação concisa dos requisitos do sistema sob o ponto de vista de dados.

- dados armazenados pelo sistema → descrevem as "coisas" do mundo real que possibilitam que os requisitos do sistema possam ser atendidos.
- relação entre dados dentro do sistema e pessoas ou coisas fora do sistema → mapa que oferece uma pista sobre como se deve organizar os dados dentro do sistema.

MUNDO REAL

Cliente

Carro

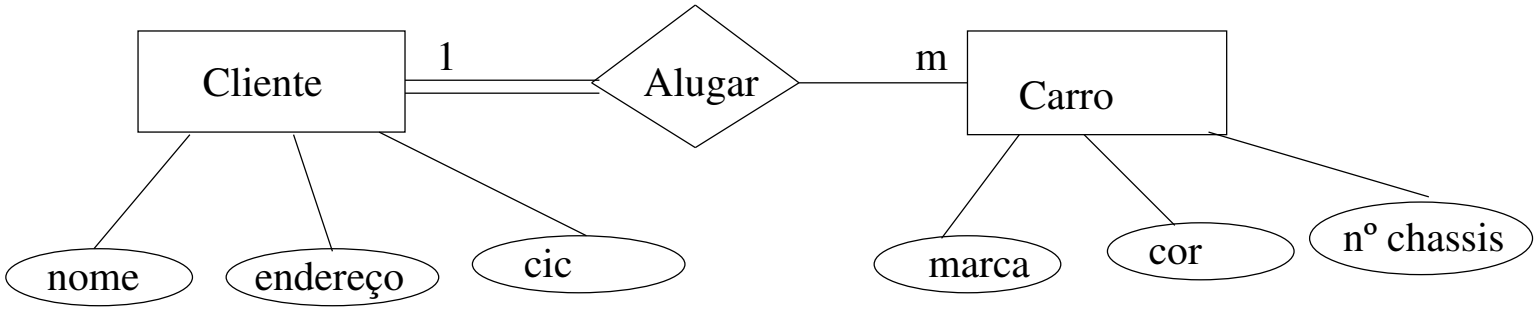


DENTRO do SISTEMA

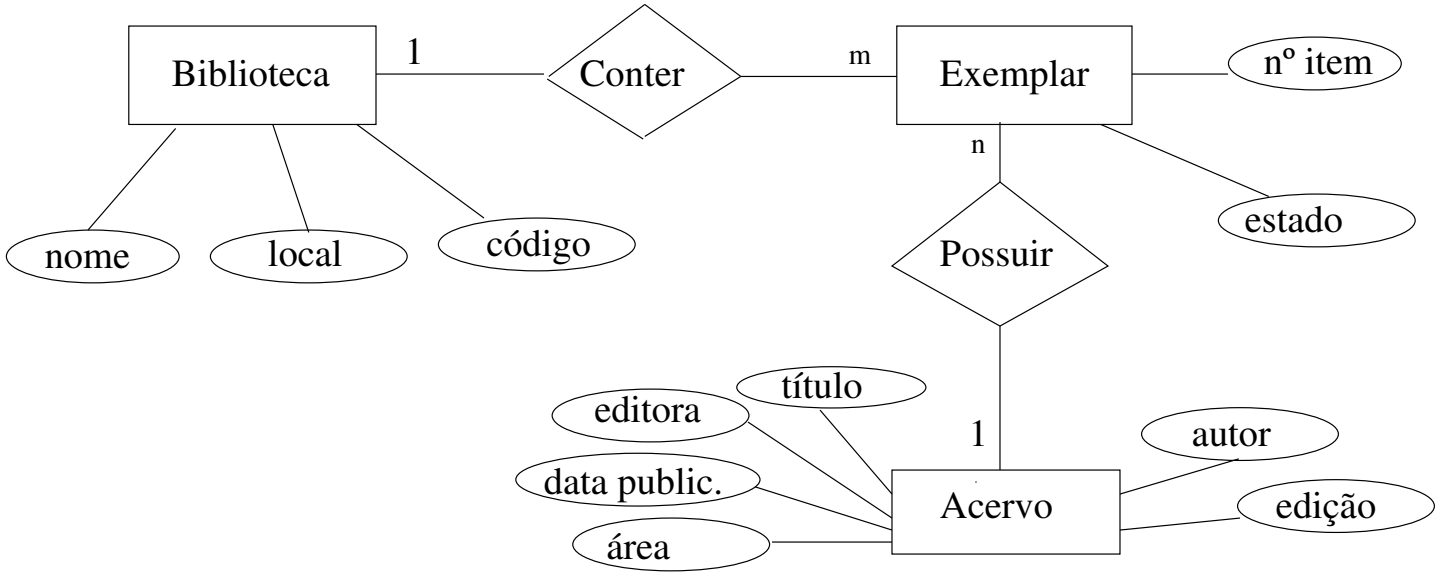
Entidade	Propriedade	Relacionamento
Cliente:	nome endereço cic	
		Alugar
Carro:	marca cor nº chassis	

- modelo utilizado para a especificação dos dados armazenados pelo sistema → MER.
- três conceitos básicos: entidades, atributos e relacionamentos.
- usa uma ferramenta gráfica (DER), que utiliza as seguintes notações:
  - *retângulo*: entidade para a qual o sistema mantém dados;
  - *elipse*: atributo (propriedade) das entidades;
  - *losango*: relacionamento entre as entidades;
  - *linha*: liga entidades a seus atributos e/ou aos relacionamentos.

- MER $\rightarrow$ especificação descritiva (propriedades das entidades em termos de seus atributos e de seus relacionamentos com outras entidades).
- pode representar outras características do mundo real, entre elas participação e cardinalidade.


Cliente
1
Alugar
m
Carro
nome
endereço
cic
marca
cor
nº chassis

- a participação de uma entidade em um relacionamento pode ser total ou parcial.
- cardinalidade: a quantidade de instâncias de uma entidade que se relacionam com instâncias de outra entidade ((1:1); (1:m); (m:m)).
- mas: como um *cic* foi formado?
- portanto: notação semiformal, pois sua sintaxe e semântica não são precisas.

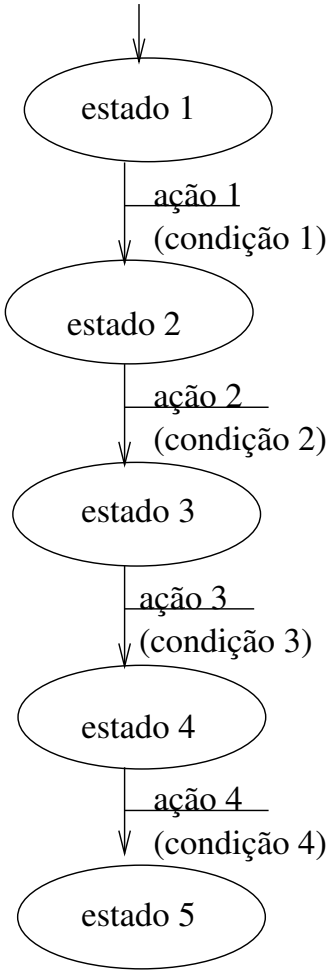

Biblioteca
1
Conter
m
Exemplar
nº item
n
estado
nome
local
código
Possuir
título
editora
1
autor
data public.
Acervo
edição
área

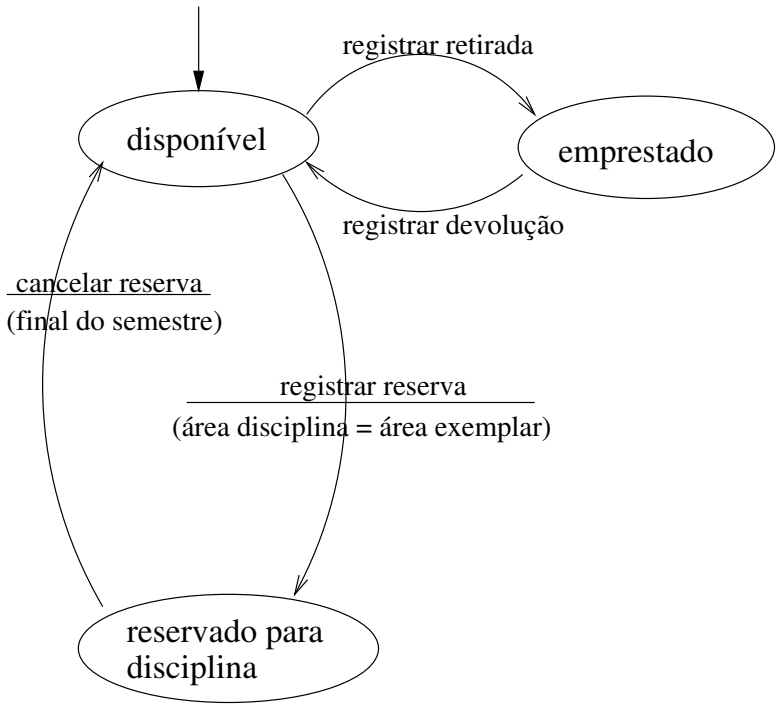
O modelo comportamental

- sistemas de software → tendem a assumir vários estados, cada um deles caracterizado por uma resposta única a um determinado conjunto de estímulos.
- análise dos estados do sistema → exige uma enumeração de seus possíveis estados e dos eventos (condições e/ou ações) que causam a mudança de estado.
- modelo do comportamento → representa os estados e os eventos que alteram esses estados.

- eventos: representados através de condições e ações para mudança de estado.
- quando a condição para a ocorrência de um evento é verdadeira $\rightarrow$ a ação correspondente é ativada.
- enquanto uma ação é realizada $\rightarrow$ o estado do componente do sistema que está sendo modelado não é observável.
- quando a ação é completada $\rightarrow$ o componente passa ao estado determinado pelo evento correspondente.
- ferramenta para descrever aspectos de comportamento $\rightarrow$ máquina de estados finito (MEF).

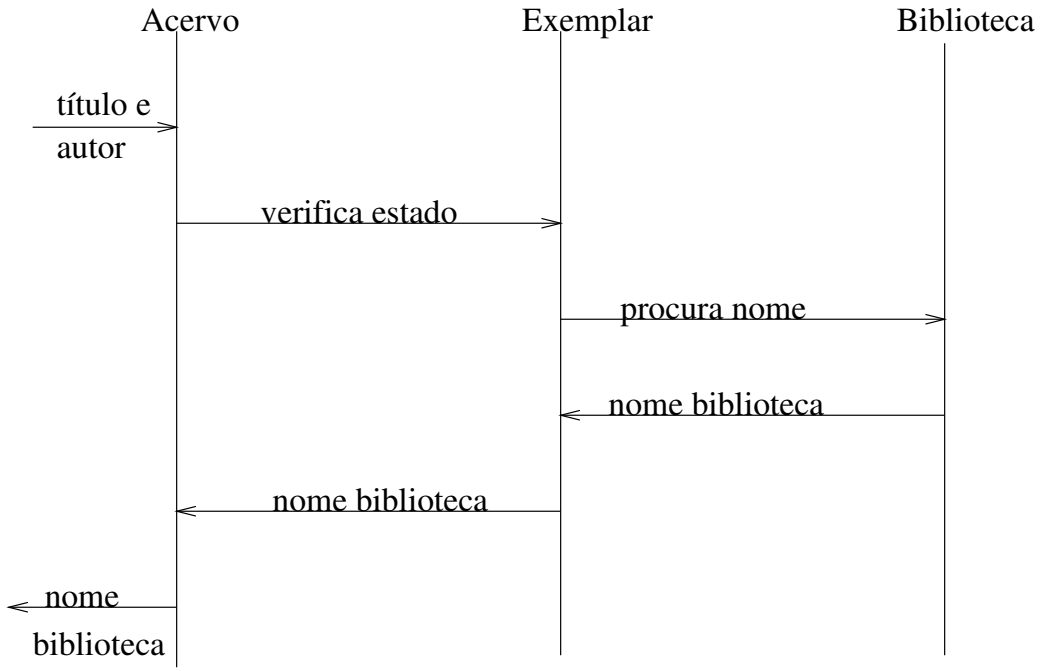
- MEF $\rightarrow$ ferramenta gráfica para especificações semiformais
- utiliza um grafo para representar o comportamento do sistema.
- descreve o sistema como um conjunto de estados que se alternam em conseqüência de algum evento modelado por dados de entrada.
- notações:
  - *elipse*: para representar os estados;
  - *seta*: para representar os eventos que causam a mudança de estado.


estado 1
ação 1
(condição 1)
estado 2
ação 2
(condição 2)
estado 3
ação 3
(condição 3)
estado 4
ação 4
(condição 4)
estado 5



- traço de eventos → modela o comportamento.
- representa cenários em sistemas orientados a objetos.
- cenários → como o sistema trabalhará quando estiver em operação.

- representa o comportamento do sistema, através dos objetos das classes envolvidas em um serviço do sistema e as interfaces.
- Notação:
  - *traço vertical*: representa as classes envolvidas no serviço;
  - *seta horizontal*: representa as mensagens trocadas.


Acervo
Exemplar
Biblioteca
título e
autor
verifica estado
procura nome
nome biblioteca
nome biblioteca
nome
biblioteca

O modelo de objetos

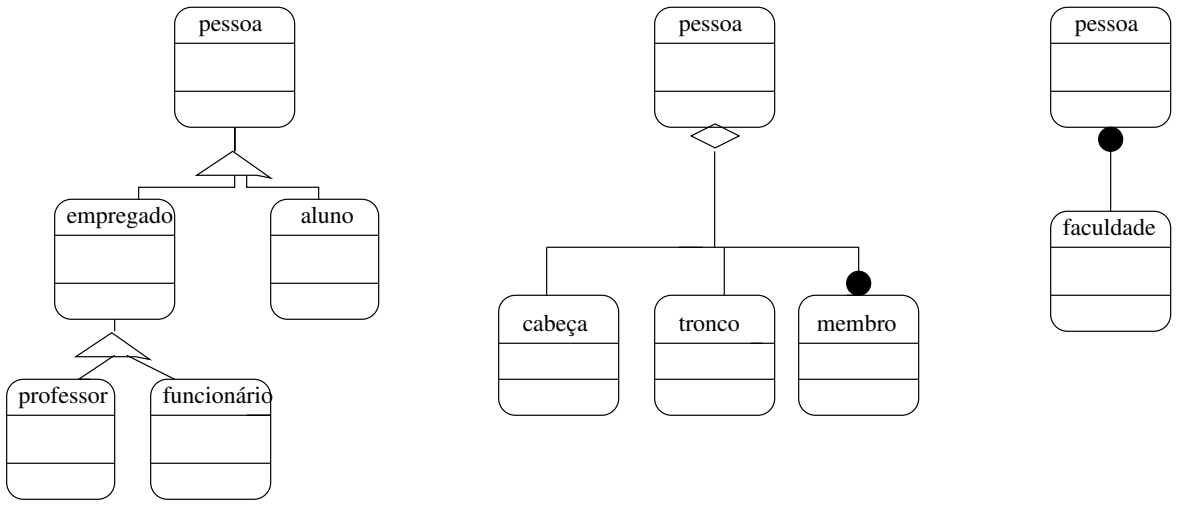
- utilizado para representar tanto dados como seu processamento.
- combina aspectos do modelo de função e do modelo de dados.
- permite que se representem a composição e a classificação de componentes do sistema (um componente é um objeto).
- o modelo não inclui detalhes de objetos individuais, e sim de uma classe de objetos representando o mundo real.

- classe de objetos $\rightarrow$ abstração sobre um conjunto de objetos que possuem atributos e serviços (operações) de classes de objetos comuns.
- representa:
  - os atributos das classes de objetos;
  - seus serviços;
  - os relacionamentos entre as classes; e
  - a utilização de serviços de um objeto por outro.
- utilizado tanto na fase de análise como na de projeto.

- fase de análise → identificadas classes que representam o domínio do problema (coisas ou conceitos do problema).
- fase de projeto → acrescentadas informações que lidam com o domínio da solução.
- utiliza um diagrama de classes de objetos para especificar o sistema.

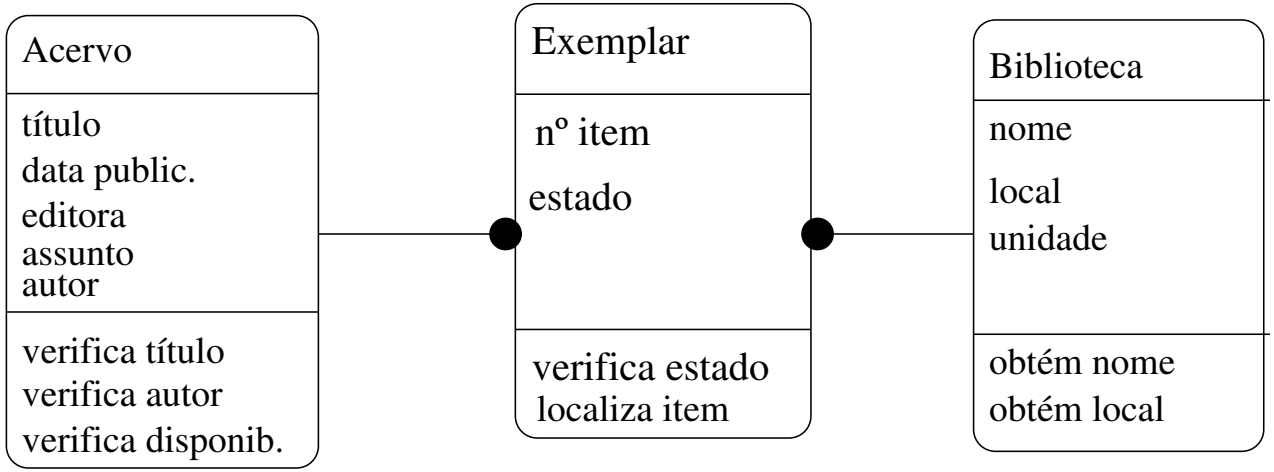
- Notação:
  - *retângulo de cantos arredondados dividido em três partes* → classes de objetos (nome da classe, lista de atributos e serviços (operações);
  - *linha ligando classes* → associações entre classes de objetos (troca de mensagen entre classes – um objeto usa serviços de outro objeto);
  - *losango* → composição de objetos de uma classe (agregação);
  - *triângulo* → classificação de objetos de uma classe (generalização/especialização).

- relações entre classes de objetos → representadas através da generalização/especialização e da agregação.
- generalização/especialização → uma classe de objetos herda todos ou alguns atributos e serviços de uma classe mais geral.
- agregação → representa a composição de objetos.



(a) (b) (c)

- serviços → operações associadas aos objetos da classe e permitem que o estado de um objeto de uma classe possa ser modificado ou observado.
- serviços são definidos para uma classe e ficam disponíveis para cada objeto dela.
- para um objeto obter um serviço de outra classe → associação entre as classes de forma que um objeto de uma classe possa enviar mensagens a objetos de outra classe que possuam o serviço desejado.


Acervo
título
data public.
editora
assunto
autor
verifica título
verifica autor
verifica disponib.
Exemplar
nº item
estado
verifica estado
localiza item
Biblioteca
nome
local
unidade
obtém nome
obtém local